

Multi-Target Tracking in Distributed Sensor Networks using Particle PHD Filters

Mark R. Balthasar, *Student Member, IEEE*, and Abdelhak M. Zoubir, *Fellow, IEEE*

Abstract—Multi-target tracking is an important problem in civilian and military applications. This paper investigates multi-target tracking in distributed sensor networks. Data association, which arises particularly in multi-object scenarios, can be tackled by various solutions. We consider sequential Monte Carlo implementations of the Probability Hypothesis Density (PHD) filter based on random finite sets. This approach circumvents the data association issue by jointly estimating all targets in the region of interest. To this end, we develop the Diffusion Particle PHD Filter (D-PPHDF) as well as a centralized version, called the Multi-Sensor Particle PHD Filter (MS-PPHDF). Their performance is evaluated in terms of the Optimal Subpattern Assignment (OSPA) metric, benchmarked against a distributed extension of the Posterior Cramér-Rao Lower Bound (PCRLB), and compared to the performance of an existing distributed PHD Particle Filter.

Index Terms—Multi-target tracking, distributed target tracking, Particle Filter, PHD Filter, autonomous sensor networks, Posterior Cramér-Rao Lower Bound

I. INTRODUCTION

THE problem of multi-target tracking (MTT) is becoming increasingly important in many military and civilian applications such as air and ground traffic control, harbor surveillance, maritime traffic control, or video communication and surveillance [1]–[3]. Distributed sensor networks offer a desirable platform for MTT applications due to their low cost and ease of deployment, their lack of a single point of failure, as well as their inherent redundancy and fault-tolerance [4]. A comprehensive overview of the state-of-the-art of distributed single-target tracking (STT) is given in [5]. Distributed versions of the Kalman Filter [5], [6] and its nonlinear, non-Gaussian counterpart, the Particle Filter (PF) [7], have been well-studied. However, they cannot be applied directly to MTT as they do not account for the problem of data association. Although there are methods such as the Joint Probabilistic Data Association Filter (JPDAF) [8] or the Multiple Hypothesis Tracker (MHT) [9] that address this problem in STT algorithms, the resource constraints in sensor networks might pose a challenge on finding suitable distributed implementations [10]. The Probability Hypothesis Density (PHD) filter [11], [12], in contrast, resorts to the concept of random finite sets (RFSs) to circumvent the problem of data association altogether.

In this work, we investigate distributed MTT in a sensor network with 1-coverage of the region of interest (ROI), i.e., the sensor nodes have non- or barely overlapping fields of view

(FOVs) and are distributed such that maximum area coverage is attained [13]. An exemplary network layout with these properties is depicted in Figure 1a). Autonomous distribution algorithms for realizing such a topology have been studied in our previous work [14]. The nodes in the network communicate with their neighbors in order to collaboratively detect and track targets in the ROI. In addition, all of the sensors are equipped with a signal processing unit, allowing them to form decisions without a fusion center. That way, the network can autonomously react to events such as the detection of an intruder without relying on a network operator. For the sake of simplicity, the network is considered to be static. However, the consideration of mobile sensor nodes would enable reactions such as target pursuit or escape.

Since the FOV and communication radius of each node are limited, a target is only seen by a subset of the network, which changes as the target moves through the ROI. Hence, at each time instant, there is an *active* and an *inactive* part of the network. The goal, thus, is to detect and observe the target in a distributed and collaborative fashion as it travels across the ROI, rather than reaching a network-wide consensus on its state and have the estimate available at each node.

In the sequel, we develop a distributed Particle PHD filter called Diffusion Particle PHD Filter (D-PPHDF), which uses neighborhood communication to collaboratively estimate and track a single-sensor PHD at each node in the active subnetwork. In addition, we formulate the Multi-Sensor Particle PHD Filter (MS-PPHDF), a centralized extension of the D-PPHDF. The performance of both algorithms is evaluated in terms of the Optimal Subpattern Assignment (OSPA) metric [15], which is calculated for the joint set of target state estimates of the active subnetwork. Furthermore, a distributed version of the Posterior Cramér-Rao Lower Bound (PCRLB) [16]–[18] – again averaged over the active subnetwork – is introduced and used as a benchmark.

Other distributed solutions for MTT in a multi-sensor setup using the PHD filter have been studied, e.g., in [19], [20], [21]. Contrary to our approach, they either assume overlapping FOVs or employ a pairwise communication scheme. The common idea, however, is to extend the single-sensor PHD filter to the multi-sensor case through communication between multiple nodes, or nodes and a fusion center. A more rigorous approach for MTT with multiple sensors is to use a multi-sensor PHD filter [22], [23], which seeks to estimate and track a single multi-sensor PHD instead of multiple single-sensor PHDs. In this work, we compare our methods to the approach in [20] (adapted to our scenario), which is also based on single-sensor PHDs. The consideration of methods based on a multi-sensor PHD will be the focus of future work.

The authors are with the Signal Processing Group, Institute of Telecommunications, Technische Universität Darmstadt, Darmstadt 64283, Germany (e-mail: balthasar@spg.tu-darmstadt.de, zoubir@spg.tu-darmstadt.de)

Manuscript submitted December 8, 2015

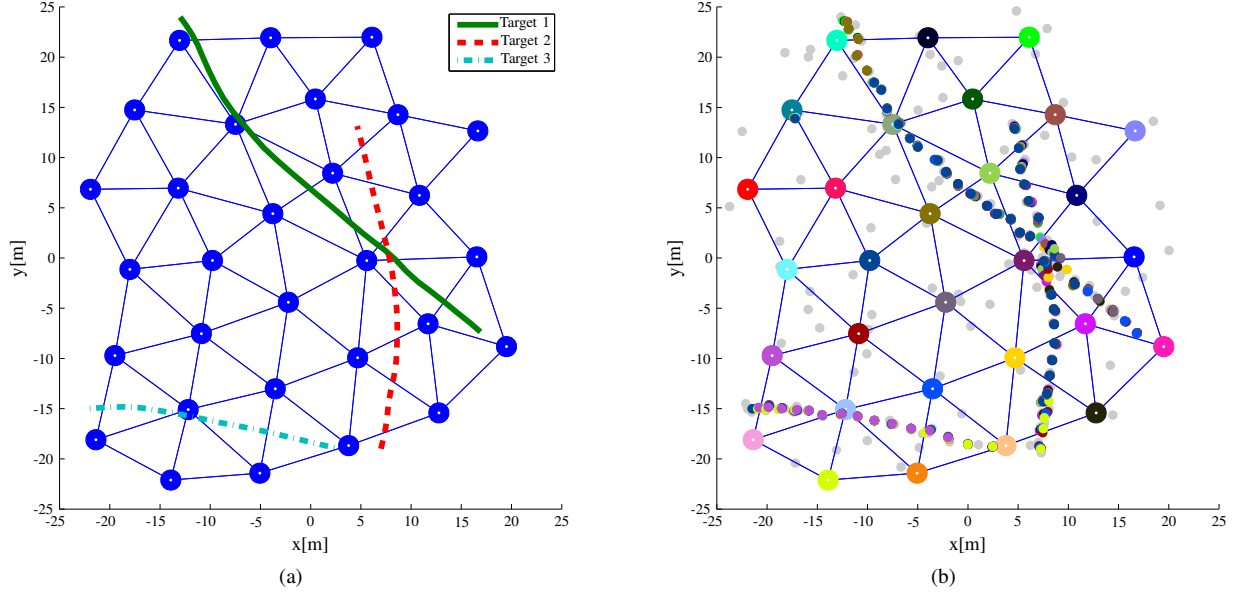


Fig. 1: (a) Distributed sensor network with 1-coverage of the region of interest and 3 exemplary target tracks. (b) Example of tracking 3 targets with the Diffusion Particle PHD Filter (D-PPHDF). The small colored dots represent the target location estimates obtained by the respective node with the same color. The light gray dots show the collective measurements obtained by all nodes in the network.

The paper is organized as follows: Section 2 presents the considered state-space model and recapitulates the theory of RFSs as well as the PHD and the PHD filter. The problem of distributed MTT is addressed in Section 3. Here, we will first detail our modification of Adaptive Target Birth (ATB) before formulating the D-PPHDF and investigate its computational complexity and communication load. In Section 4, the MS-PPHDF is developed and analyzed in terms of computational complexity and communication load. Section 5 is dedicated to simulations. First, the Distributed Posterior Cramér-Rao Lower Bound (DPCRLB) is introduced. Then, we present the simulation setup and discuss our results. Finally, a conclusion is given in Section 6.

II. MODELS AND THEORY

A. State-Space Model

A linear state-space model is considered for each target at time instant $i \geq 0$. The target state vector $\mathbf{s}^{\text{tgt}}(i) = [\mathbf{x}^{\text{tgt}}(i), \dot{\mathbf{x}}^{\text{tgt}}(i)]^T$ contains the target location vector \mathbf{x}^{tgt} as well as the velocity vector $\dot{\mathbf{x}}^{\text{tgt}}$. For the sake of simplicity, we restrict ourselves to a 2D-environment. The target state evolves according to the state equation [24]:

$$\mathbf{s}^{\text{tgt}}(i) = \mathbf{F}(i)\mathbf{s}^{\text{tgt}}(i-1) + \mathbf{G}(i)\mathbf{n}^{\text{tgt}}(i). \quad (1)$$

The matrices \mathbf{F} and \mathbf{G} as well as the vector \mathbf{n}^{tgt} will be explained shortly. Node k obtains a measurement \mathbf{z}_k of the target location as given by the measurement equation [24]:

$$\mathbf{z}_k(i) = \mathbf{H}_k(i)\mathbf{s}^{\text{tgt}}(i) + \boldsymbol{\nu}_k^{\text{tgt}}(i), \quad k \in \mathcal{M} \quad (2)$$

with $\mathcal{M} = \{m \in \{1, \dots, N\} \mid \|\mathbf{x}_m(i) - \mathbf{x}^{\text{tgt}}(i)\|_2 \leq R_{\text{sen}}\}$ denoting the set of all nodes m that are located such that the

Euclidean distance $\|\mathbf{x}_m(i) - \mathbf{x}^{\text{tgt}}(i)\|_2$ between their location \mathbf{x}_m and the target location \mathbf{x}^{tgt} is not greater than their sensing radius R_{sen} . Note that N is the total number of nodes in the network. Furthermore, $\mathbf{n}^{\text{tgt}}(i) \sim \mathcal{N}(\mathbf{0}_{2,1}, \mathbf{Q}(i))$ and $\boldsymbol{\nu}_k^{\text{tgt}}(i) \sim \mathcal{N}(\mathbf{0}_{2,1}, \mathbf{R}_k(i))$ denote the state and measurement noise processes, respectively, with the zero-mean vector $\mathbf{0}_{2,1} = [0, 0]^T$. Both noise processes are spatially and temporally white, as well as uncorrelated with the initial target state $\mathbf{s}^{\text{tgt}}(0)$ and each other for all i . For the sake of simplicity, we choose a time-invariant measurement noise covariance matrix

$$\mathbf{R}_k(i) = \mathbf{R}_k = \sigma_r^2 \mathbf{I}_2, \quad (3)$$

where σ_r^2 is the variance of each component of the measurement noise and \mathbf{I}_n denotes the identity matrix of size n .

In target tracking, the model matrices are usually chosen to be time-invariant and given by [24]

$$\mathbf{F} = \begin{bmatrix} \mathbf{I}_2 & \Delta i \mathbf{I}_2 \\ \mathbf{0}_{2,2} & \mathbf{I}_2 \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} \frac{\Delta i^2}{2} \mathbf{I}_2 \\ \Delta i \mathbf{I}_2 \end{bmatrix}, \quad \mathbf{Q} = \sigma_q^2 \mathbf{I}_2, \quad (4)$$

where $\mathbf{0}_{2,2}$ is the 2×2 zero matrix. Furthermore, Δi is the time step interval in seconds with which the state-space model progresses. In addition, σ_q^2 denotes the variance of a state noise component. We assume that the sensor nodes only obtain location information, for instance, through distance and bearing measurements from which an estimate of the target location can easily be calculated. This yields a general measurement matrix \mathbf{H}_k of the form

$$\mathbf{H}_k = [\mathbf{I}_2 \quad \mathbf{0}_{2,2}]. \quad (5)$$

B. Random Finite Sets (RFSs)

A random finite set (RFS) is an unordered finite set that is random in the number of its elements as well as in

their values [25]. Therefore, RFSs are a natural choice for representing the multi-target states and measurements in MTT: the state and measurement vectors of all targets are collected in corresponding RFSs [26], [27]. Given the realization Ξ_{i-1} of the RFS Ξ_{i-1} at time instant $i-1$, the multi-target state of our tracking problem can be described by the RFS Ξ_i according to

$$\Xi_i = \mathcal{S}_i(\Xi_{i-1}) \cup \mathcal{B}_i, \quad (6)$$

where the survival set $\mathcal{S}_i(\Xi_{i-1})$ denotes the RFS of targets that already existed at time step $i-1$ and have not exited the ROI, i.e., the region covered by the sensor network, in the transition to time step i . In addition, the birth set \mathcal{B}_i is the RFS of new targets that spontaneously appear at the border of the ROI at time instant i [1], [12], [27]. Note that the statistical behavior of Ξ_i can be described by the conditional probability $f_{i|i-1}(\Xi_i|\Xi_{i-1})$.

The multi-target measurement model is given by the RFS Σ_i as

$$\Sigma_i = \Theta_i(\Xi_i) \cup \mathcal{C}_i(\Xi_i), \quad (7)$$

where $\Theta_i(\Xi_i)$ is the RFS of measurements generated by Ξ_i . In addition, the RFS $\mathcal{C}_i(\Xi_i)$ represents clutter or false alarms. Given a realization Σ_i of Σ_i , the statistical behavior of the RFS Σ_i is described by the conditional probability $f_i(\Sigma_i|\Xi_i)$.

C. The Probability Hypothesis Density (PHD)

In analogy to the single-target case, the optimal Bayesian filter for MTT recursively propagates the multi-target posterior $f_{i|i}(\Xi_i|\Sigma_{0:i})$ over time, according to

$$f_{i|i}(\Xi_i|\Sigma_{0:i}) = \frac{f_i(\Sigma_i|\Xi_i)f_{i|i-1}(\Xi_i|\Sigma_{0:i-1})}{\int f_i(\Sigma_i|\Xi)f_{i|i-1}(\Xi|\Sigma_{0:i-1})\mu_s(d\Xi)} \quad (8)$$

$$f_{i|i-1}(\Xi_i|\Sigma_{0:i-1}) = \int f_{i|i-1}(\Xi_i|\Xi)f_{i-1|i-1}(\Xi|\Sigma_{0:i-1})\mu_s(d\Xi), \quad (9)$$

where μ_s is a dominating measure as described in [27]. This approach requires the evaluation of multiple integrals, which makes it even more computationally challenging than its single-target counterpart. A common solution is to find a set of statistics, e.g., the moments of first or second order, which yield a good approximation of the posterior, and propagate them instead [1].

The Probability Hypothesis Density (PHD) $D_{i|i}(\mathbf{s}(i)|\Sigma_{0:i})$ is an indirect first-order moment of $f_{i|i}(\Xi_i|\Sigma_{0:i})$ [28]. It is given by the following integral [12], [29]:

$$\begin{aligned} D_{i|i}(\mathbf{s}(i)|\Sigma_{0:i}) &= \sum_{\mathbf{s}_n^{\text{tgt}}(i) \in \Xi_i} \int \delta(\mathbf{s}(i) - \mathbf{s}_n^{\text{tgt}}(i)) f_{i|i}(\mathbf{s}(i)|\Sigma_{0:i}) d\mathbf{s}(i), \end{aligned} \quad (10)$$

where $\int f(Y)\delta Y$ denotes a set integral.

The PHD has the following two properties [28]:

- 1) The expected number of targets $\hat{N}_{\text{tgt}}(i)$ at time step i is obtained by integrating the PHD according to

$$\hat{N}_{\text{tgt}}(i) = \int D_{i|i}(\mathbf{s}(i)|\Sigma_{0:i}) d\mathbf{s}(i). \quad (11)$$

This is in contrast to probability density functions (PDFs), which always integrate to 1.

- 2) Estimates of the individual target states can be found by searching for the $\lfloor \hat{N}_{\text{tgt}} \rfloor$ highest peaks of the PHD, where $\lfloor \cdot \rfloor$ denotes rounding to the nearest integer.

Because of these two properties, the number of targets as well as their states can be estimated independently at each time step without any knowledge of their identities. That way, the data association issue is avoided. However, this also means that PHD Filters cannot deliver the continuous track of a specific target. If continuous tracks are required, an additional association step has to be performed. Two possible association algorithms for track continuity can be found in [11].

D. The PHD Filter

The PHD Filter is an approach for recursively propagating the PHD $D_{i|i}(\mathbf{s}(i)|\Sigma_{0:i})$ at time step i given measurements up to time step i over time. If the RFS Ξ is Poisson-distributed, then its PHD is equal to its intensity function and is, hence, a sufficient statistic [12]. In this case, the PHD recursion is given by the following prediction and update equations [12]:

$$\begin{aligned} D_{i|i-1}(\mathbf{s}(i)|\Sigma_{0:i-1}) &= b_i(\mathbf{s}(i)) + \int p_S(\mathbf{s}(i-1)) \times \\ &\quad f_{i|i-1}(\mathbf{s}(i)|\mathbf{s}(i-1)) D_{i-1|i-1}(\mathbf{s}(i-1)|\Sigma_{0:i-1}) d\mathbf{s}(i-1) \end{aligned} \quad (12)$$

$$\begin{aligned} D_{i|i}(\mathbf{s}(i)|\Sigma_{0:i}) &= \left[1 - p_D + \right. \\ &\quad \left. \sum_{\mathbf{z} \in \Sigma_i} \frac{p_D f_i(\mathbf{z}|\mathbf{s}(i))}{\lambda_{\text{FA}} c_{\text{FA}}(\mathbf{z}) + p_D \int f_i(\mathbf{z}|\mathbf{s}(i)) D_{i|i-1}(\mathbf{s}(i)|\Sigma_{0:i-1}) d\mathbf{s}(i)} \right] \\ &\quad \times D_{i|i-1}(\mathbf{s}(i)|\Sigma_{0:i-1}) \end{aligned} \quad (13)$$

Note that $b_i(\mathbf{s}(i))$ is the PHD of the birth set \mathcal{B}_i of new targets appearing at time step i . In addition, $p_S(\mathbf{s}(i-1))$ denotes the probability that a target survives the transition from time step $i-1$ to i . The probability of survival depends on the previous state $\mathbf{s}(i-1)$ because a target that is close to the border of the ROI and has a velocity vector pointing away from it is unlikely to be present at time step i . Furthermore, $f_{i|i-1}(\mathbf{s}(i)|\mathbf{s}(i-1))$ and $f_i(\mathbf{z}|\mathbf{s}(i))$ denote the transition probability and the likelihood, respectively. The probability of detection p_D is constant since it is assumed that all targets can be detected if the ROI is covered. The term $\lambda_{\text{FA}} c_{\text{FA}}(\mathbf{z})$ represents Poisson-distributed false-alarms due to clutter, where λ_{FA} is the false alarm parameter, which is distributed according to its probability density function $c_{\text{FA}}(\mathbf{z})$.

III. DISTRIBUTED MULTI-TARGET TRACKING

In this section we introduce the Diffusion Particle PHD Filter (D-PPHDF), a distributed Particle Filter implementation of the PHD Filter for performing MTT in a sensor network without a fusion center. Before diving into the algorithm, we briefly review the concept of Adaptive Target Birth (ATB) and discuss the modification we applied in the D-PPHDF.

A. Adaptive Target Birth (ATB)

Standard formulations of the PHD Filter consider the PHD $b_i(\mathbf{s}(i))$ of the birth set \mathcal{B}_i to be known a priori [30]. For typical tracking applications such as air surveillance, this is a reasonable assumption since new targets should appear at the border of the ROI given continuous observation. An alternative is to make the target birth process adaptive and measurement-driven as suggested in [30], [31]. To this end, the PHD – and consequently the set of particles and weights approximating it in a Particle Filter implementation – is split into two densities corresponding to persistent objects, which have survived the transition from time step $i - 1$ to i , and newborn objects, respectively.

In [30], [31], the PHD of newborn objects is approximated by randomly placing N_P new particles around each target measurement, with N_P denoting the number of particles per target. We improve upon this approach by only considering measurements with no noticeable impact on any persistent particle weight, as these may indicate the appearance of a new target. That way, the number of newborn particles is further reduced and a possible overlap between persistent and newborn PHD is avoided. With the transition to time step $i + 1$, the newborn particles become persistent. Furthermore, we perform the ATB step towards the end of each iteration of the algorithm and only consider the particles representing the persistent PHD in the prediction, weighting, and resampling steps. Hence, the update equation (13) does not have to be modified as in [30], [31].

While ATB delays the tracking algorithm by one time step, it is much more efficient as it only places new particles in regions in which a target is likely to be found. In addition, there is no need for an explicit initialization step since the first incoming target will trigger the deployment of a newborn particle cloud around its corresponding measurement.

B. The Diffusion Particle PHD Filter (D-PPHDF)

The proposed Diffusion Particle PHD Filter (D-PPHDF) is an extension of the single-sensor Particle PHD Filter (PPHDF) [11], [26], [32] for the multi-sensor case. Furthermore, it relies on ATB for a more efficient target detection. The communication scheme we employ to exchange measurements and estimates between nodes is inspired by the two-step communication used in the context of Diffusion Adaptation [33]. However, the algorithm does not rely on least-mean-squares or any other kind of adaptive filter. First, each node k in the active part of the network obtains an intermediate estimate of the states of the targets present, i.e., of the PHD of persistent targets – represented by the set $\left\{ \mathbf{s}_{k,\text{pers}}^p(i), w_{k,\text{pers}}^p(i) \right\}_{p=1}^{N_{k,\text{pers}}(i)}$ of persistent particles with corresponding weights – based on neighborhood measurements. In other words, every active node runs a separate PPHDF with access to measurements from its neighborhood \mathcal{N}_k , defined as

$$\mathcal{N}_k = \{l \in \{1, \dots, N\} \mid \|\mathbf{x}_l - \mathbf{x}_k\|_2 \leq R_{\text{com}}\}, \quad k = 1, \dots, N, \quad (14)$$

where R_{com} denotes the communication radius. Second, each active node combines the intermediate estimates from its

neighborhood to a final, collaborative estimate. To this end, the persistent particle sets of all neighbors are merged into a collective set $\left\{ \mathbf{s}_{k,\text{coll}}^p(i), w_{k,\text{coll}}^p(i) \right\}_{p=1}^{N_{k,\text{coll}}(i)}$ of persistent neighborhood particles and corresponding weights before the clustering step, with $N_{k,\text{coll}}(i)$ denoting the number of collective persistent neighborhood particles. In the sequel, we will look at the individual steps of the D-PPHDF in more detail:

- *Merging*: The sets $\left\{ \mathbf{s}_{k,\text{coll}}^p(i-1), w_{k,\text{coll}}^p(i-1) \right\}_{p=1}^{N_{k,\text{coll}}(i-1)}$ and $\left\{ \mathbf{s}_{k,\text{new}}^p(i-1), w_{k,\text{new}}^p(i-1) \right\}_{p=1}^{N_{k,\text{new}}(i-1)}$ consist of the collective persistent neighborhood particles and newborn particles of node k , $\mathbf{s}_{k,\text{coll}}^p(i-1)$ and $\mathbf{s}_{k,\text{new}}^p(i-1)$, respectively, at time step $i-1$ with their respective weights $w_{k,\text{coll}}^p(i-1)$ and $w_{k,\text{new}}^p(i-1)$. These sets are merged to become the total set $\left\{ \mathbf{s}_{k,\text{tot}}^p(i), w_{k,\text{tot}}^p(i) \right\}_{p=1}^{N_{k,\text{tot}}(i)}$ of particles and weights of node k at time step i . Here, $N_{k,\text{tot}}(i)$ is the total number of particles of node k at time step i , which is given by

$$N_{k,\text{tot}}(i) = N_{k,\text{coll}}(i-1) + N_{k,\text{new}}(i-1), \quad (15)$$

with $N_{k,\text{coll}}(i-1)$ and $N_{k,\text{new}}(i-1)$ denoting the respective number of persistent neighborhood and newborn particles at the previous time step. Note that since the sets of particles and weights represent PHDs, merging the sets corresponds to summing these PHDs.

- *Predicting*: Each particle is propagated through the system model to become a persistent particle. The system model is assumed to be the same for each target and given by Equation (1). Since the process noise is captured by the spread of the particle cloud, the respective term can be removed from the equation, yielding

$$\mathbf{s}_{k,\text{pers}}^p(i) = \mathbf{F} \mathbf{s}_{k,\text{tot}}^p(i), \quad p = 1, \dots, N_{k,\text{tot}}(i). \quad (16)$$

The corresponding weights are multiplied with the probability of survival p_S , which is assumed to be constant for the sake of simplicity¹, according to

$$w_{k,\text{pers}}^p(i|i-1) = p_S w_{k,\text{tot}}^p(i), \quad p = 1, \dots, N_{\text{pers}}(i). \quad (17)$$

The prediction of particles and weights corresponds to the second term in Equation (12).

- *Measuring & Broadcasting (1)*: The sensor nodes obtain measurements of the targets and forward them to their neighbors.
- *Weighting*: The persistent particle weights of node k are updated by applying a weighting step corresponding to Equation (13) iteratively for each neighbor. Using the product operator, this weighting step can be compactly denoted as

¹A constant probability of survival p_S is a reasonable assumption if the targets move relatively slowly with respect to the observation time and the size of the ROI.

$$w_{k,\text{pers}}^p(i) = \prod_{l \in \mathcal{N}_k} \left[1 - p_D + \sum_{\mathbf{z}_j \in \Sigma_i^l} w_{k,j,\text{update}}^p(i) \right] w_{k,\text{pers}}^p(i|i-1), \quad (18)$$

with

$$w_{k,j,\text{update}}^p(i) = \frac{p_D f_i(\mathbf{z}_j | \mathbf{x}^p(i))}{\lambda_{\text{FA}} c_{\text{FA}}(\mathbf{z}_j) + \mathcal{L}(\mathbf{z}_j)}, \quad (19)$$

where Σ_i^l is the set of measurements obtained by node l and $\mathcal{L}(\mathbf{z}_j)$ is calculated as

$$\mathcal{L}(\mathbf{z}_j) = \sum_{q=1}^{N_{k,\text{pers}}^p(i)} p_D f_i(\mathbf{z}_j | \mathbf{x}^q(i)) w_{k,\text{pers}}^q(i|i-1). \quad (20)$$

Note that $f_i(\mathbf{z}_j | \mathbf{x}^p(i))$ is the likelihood and $\mathbf{x}^p(i)$ is the location vector of particle p .

Afterwards, each node k obtains the set $\Sigma_{i,\text{cand}}^k$ of candidate measurements, i.e., measurements that are not responsible for the highest weighting of any persistent particle, to be used in the ATB step later on. The set $\Sigma_{i,\text{cand}}^k$ is found according to

$$\Sigma_{i,\text{cand}}^k = \Sigma_i^k \setminus \left\{ \mathbf{z}_{m_p} \mid m_p = \arg \max_j w_{k,j,\text{update}}^p(i), p = 1, \dots, N_{k,\text{pers}}(i) \right\}. \quad (21)$$

- *Resampling*: Each node k calculates its own expected number of targets $\hat{N}_{k,\text{tgt}}(i)$ from its total persistent particle mass according to

$$\hat{N}_{k,\text{tgt}}(i) = \left\lfloor \sum_{p=1}^{N_{k,\text{tot}}(i)} w_{k,\text{pers}}^p(i) \right\rfloor. \quad (22)$$

Consequently, the number of persistent particles of node k is updated as

$$N_{k,\text{pers}}(i) = \hat{N}_{k,\text{tgt}}(i) N_P. \quad (23)$$

Furthermore, the set of persistent particles of node k has to be resampled by drawing $N_{k,\text{pers}}(i)$ particles with replacement from it. Note that the probability of drawing particle p is given by $\frac{w_{k,\text{pers}}^p(i)}{\hat{N}_{k,\text{tgt}}(i)}$ since the weights do not sum to unity. Then, the weights are reset to equal values as

$$w_{k,\text{pers}}^p(i) = \frac{\hat{N}_{k,\text{tgt}}(i)}{N_{k,\text{pers}}(i)}, \quad p = 1, \dots, N_{k,\text{pers}}(i). \quad (24)$$

- *Broadcasting (2)*: Every node k transmits its set of resampled persistent particles and weights $\left\{ \mathbf{s}_{k,\text{pers}}^p(i), w_{k,\text{pers}}^p(i) \right\}_{p=1}^{N_{k,\text{pers}}(i)}$ to its neighbors.
- *Clustering*: Each node k forms a collective set of persistent neighborhood particles $\mathbf{s}_{k,\text{coll}}^p(i)$ and corresponding

weights $w_{k,\text{Nh}}^p(i)$ according to

$$\left\{ \mathbf{s}_{k,\text{coll}}^p(i), w_{k,\text{coll}}^p(i) \right\}_{p=1}^{N_{k,\text{coll}}(i)} = \bigcup_{l \in \mathcal{N}_k} \left\{ \mathbf{s}_{l,\text{pers}}^p(i), w_{l,\text{pers}}^p(i) \right\}_{p=1}^{N_{l,\text{pers}}(i)}, \quad (25)$$

with

$$N_{k,\text{coll}}(i) = \sum_{l \in \mathcal{N}_k} N_{l,\text{pers}}(i) \quad (26)$$

denoting the number of collective persistent neighborhood particles of node k . As in the *merging* step, this corresponds to summing the corresponding PHDs to obtain an updated single-sensor PHD with a probability distribution reflecting the information of the entire neighborhood of node k . Note that the PHDs might not be independent if a target is detected by more than one neighbor. However, this is not a problem since merging the particle sets simply results in the respective target being represented by more particles. Hence, node k will be able to estimate the corresponding location more accurately.

The estimated target states are found by clustering the collective persistent particles. Since the expected number of targets $\hat{N}_{l,\text{tgt}}(i)$, $l \in \mathcal{N}_k$ might be different for each neighbor, we resort to *hierarchical clustering of the single-linkage type* [34]. Here, the sum of the expected number of targets over the neighborhood can serve as an upper bound for the number of clusters. Note, however, that if two targets are close to each other, clustering algorithms might not be able to resolve both targets correctly.

- *Roughening*: A roughening step is performed to counter sample impoverishment [35]. To this end, an independent jitter $\mathbf{s}^j(i)$ is added to every resampled particle. Each component $s_c^j(i)$, $c = 1, \dots, d$ of the jitter with dimensionality d is sampled from the Gaussian distribution $\mathcal{N}(0, (\sigma_c^j(i))^2)$. The component-wise standard deviation of the jitter is given by

$$\sigma_c^j(i) = K E_c N_{k,\text{coll}}(i)^{-1/d}, \quad (27)$$

where E_c is the interval length between the maximum and minimum samples of the respective component. To avoid evaluating E_c separately for each particle cluster, it is assigned an empirically found constant value.² Note that $d = 4$ since the dimensionality of the jitter vector $\mathbf{s}^j(i)$ and the particle state vector $\mathbf{s}^p(i)$ have to coincide. In addition, K is a tuning constant, which controls the spread of the particle cloud.

- *Adaptive Target Birth*: N_P new particles are placed randomly around each candidate measurement $\mathbf{z}_j \in \Sigma_{i,\text{cand}}^k$ leading to a total number of $N_{k,\text{new}}(i) = N_P \cdot |\Sigma_{i,\text{cand}}^k|$ newborn particles for node k . Every newborn particle is

²Since the noise variances as well as the network topology are fixed, the true value of E_c will not change significantly over time and between clusters, so this is a valid simplification.

associated with a weight that is chosen according to

$$w_{k,\text{new}}^p(i) = \frac{p_B}{N_{k,\text{new}}^p(i)}, \quad p = 1, \dots, N_{k,\text{new}}(i), \quad (28)$$

where p_B is the probability of birth. Depending on the application, p_B can depend on time as well as on the location of the respective particle. For simplicity, the probability that a new target enters the ROI is assumed to be equal for all locations in the birth region over time. The target birth process corresponds to the first term in Equation (12).

Figure 1b) shows an example of tracking three targets, which move along the deterministic tracks depicted in Figure 1a), using the D-PPHDF. Note that each small colored dot corresponds to a target location estimate obtained by the respective node with the same color while the light grey dots represent the collective measurements from all nodes. From this illustration, the following properties of the D-PPHDF are apparent: First, the algorithm only delivers separate location estimates – represented by the small colored dots – for each time instant rather than continuous tracks, which – as mentioned before – is a common property of PHD filters. Second, the network as a whole would be able to correctly track all three targets, while a single node only obtains the locally relevant subtracks of the targets in its vicinity. Third, the employed two-step communication theme is able to extend the vicinity of a node far beyond its own sensing radius of $R_{\text{sen}} = 6$ m. This can, for instance, be seen from the fact that the lime-green node located at $[-14, -23]$ is able to obtain location estimates of target 2, which enters the ROI from the south. Finally, Figure 1b) also illustrates the resolution problem of clustering. When targets 1 and 2, which enter the ROI from the north and the south, respectively, cross paths, the nodes in their vicinity see them as just one target. This leads to an aggregation of target location estimates around $[9, 0]$.

The pseudo-code of the D-PPHDF is given in Table II.

C. Computational Complexity and Communication Load

In this section we take a look at the computational complexity and the communication load the D-PPHDF imposes on each node in the active subnetwork. The following steps are performed at every time instant i but time dependency is omitted for simplicity. Note that each of the steps scales with the number of active nodes when considering the computational complexity of the network as a whole.

- *Prediction*: The prediction step described by Equations (16) and (17) is performed for each particle at every active node. Hence, it scales with the number of particles $N_{k,\text{tot}}$ and the dimensionality d of the particle vectors. In order to obtain a tractable expression for the computational complexity, we assume each node to have the same number of particles N_{tot} .
 $\Rightarrow \mathcal{O}(N_{\text{tot}}d)$
- *Weighting*: Each particle is updated in the weighting step given by Equations (18)-(21). The weight update as well as the designation of candidate measurements for ATB depends on the neighborhood size $|\mathcal{N}_k|$ of node k and the

number of measurements $|\Sigma^l|$ of each of its neighbors l . For tractability reasons, we assume each node to have the same number of neighbors N_{nb} and to obtain the same number of measurements N_{meas} .

$$\Rightarrow \mathcal{O}(N_{\text{tot}}N_{\text{nb}}N_{\text{meas}})$$

- *Resampling*: The estimation of the number of targets and the resampling step in Equations (22)-(24) are linear in the number of particles used for the calculation [36]. For the sake of simplicity, we assume each active node to have the same estimate of the number of targets N_{tgt} .
 $\Rightarrow \mathcal{O}(N_{\text{tot}} + N_{\text{active}}N_{\text{tgt}}N_P)$
- *Clustering*: The complexity of single-linkage clustering is cubic in the number of particles, i.e., in the number of neighbors N_{nb} of each node, the estimated number of targets N_{tgt} , the number of particles per target N_P , and the dimensionality d of the particles [37].
 $\Rightarrow \mathcal{O}((N_{\text{nb}}N_{\text{tgt}}N_Pd)^3)$
- *Roughening*: Roughening (Equation (27)) is performed for every collective particle and is linear in the dimensionality of the particles.
 $\Rightarrow \mathcal{O}(N_{\text{nb}}N_{\text{tgt}}N_Pd)$
- *Adaptive Target Birth*: The birth process depends on the number of particles per target N_P as well as the number of candidate measurements N_{cand} , which is assumed equal for each active node to ensure tractability.
 $\Rightarrow \mathcal{O}(N_PN_{\text{cand}})$

As far as the communication load is concerned, the D-PPHDF requires the broadcasting of measurements, i.e., 2 scalars per measurement, over the neighborhood in the first broadcasting step. In the second step, the sets of particles and weights, i.e., 5 scalars per particle, are transmitted. Clearly, the communication load strongly depends on the number of nodes in the network, or more precisely the number of active nodes and the size of their respective neighborhood. As an extension of the D-PPHDF, one could think of changing the second broadcasting step and transmit Gaussian Mixture Model representations – instead of the actual particles and weights – that will be resampled at the receiver node. That way, communication load could be reduced to transmitting only a few scalars in the second broadcasting step at the cost of estimation accuracy and additional computational complexity. However, a thorough treatment of this extension is beyond the scope of this work.

IV. CENTRALIZED MULTI-TARGET TRACKING

Having presented the D-PPHDF as a distributed solution for MTT in a sensor network, we propose the centralized counterpart to our approach in the sequel.

A. The Multi-Sensor Particle PHD Filter (MS-PPHDF)

The proposed Multi-Sensor Particle PHD Filter (MS-PPHDF) is a centralized, multi-sensor PPHDF that relies on a fusion center with access to the measurements of all nodes in the network. It is based on the formulation of the single-sensor PPHDF in [11], [26], [32] but with an extended measurement set comprising the measurements of the entire network. Hence, one might obtain more than one

measurements per target – a change to the typical assumption in target tracking that each target produces *at most* one measurement [8]. To account for this change, we add a pre-clustering step before the weighting step and normalize the weight update accordingly. A similar partitioning of the measurement set is used in extended target tracking, where a sensor can receive multiple target reflections due to the target's physical extent [38], [39].

In the following, we will look at the individual steps of the algorithm in more detail:

- **Merging:** The sets $\{s_{\text{pers}}^p(i-1), w_{\text{pers}}^p(i-1)\}_{p=1}^{N_{\text{pers}}(i-1)}$ and $\{s_{\text{new}}^p(i-1), w_{\text{new}}^p(i-1)\}_{p=1}^{N_{\text{new}}(i-1)}$ consist of the persistent and newborn particles, $s_{\text{pers}}^p(i-1)$ and $s_{\text{new}}^p(i-1)$, respectively, at time step $i-1$ with their respective weights $w_{\text{pers}}^p(i-1)$ and $w_{\text{new}}^p(i-1)$. These sets are merged to become the total set $\{s_{\text{tot}}^p(i), w_{\text{tot}}^p(i)\}_{p=1}^{N_{\text{tot}}(i)}$ of particles and weights at time step i . Here, $N_{\text{tot}}(i)$ is the total number of particles at time step i , which is given by

$$N_{\text{tot}}(i) = N_{\text{pers}}(i-1) + N_{\text{new}}(i-1), \quad (29)$$

with $N_{\text{pers}}(i-1)$ and $N_{\text{new}}(i-1)$ denoting the respective number of persistent and newborn particles at the previous time step.

- **Predicting:** As in the D-PPHDF, each particle is propagated through the system model according to

$$s_{\text{pers}}^p(i) = \mathbf{F} s_{\text{tot}}^p(i), \quad p = 1, \dots, N_{\text{pers}}(i) = N_{\text{tot}}(i) \quad (30)$$

to become a persistent particle. The corresponding weights are multiplied with the probability of survival p_S as

$$w_{\text{pers}}^p(i|i-1) = p_S w_{\text{tot}}^p(i), \quad p = 1, \dots, N_{\text{pers}}(i). \quad (31)$$

- **Measuring:** The sensor nodes obtain measurements of the targets.
- **Pre-Clustering:** Since there might be more than one measurement per target, the measurements of the entire network are pre-clustered before the weighting step and each measurement is assigned a label $C(z)$ that reflects the cardinality of its own cluster. This can be done, for instance, using single-linkage clustering [34]. The clustering is based on the distance between measurements, i.e., spatially close measurements are assumed to stem from the same target. Hence, when two or more targets are too close to each other, cardinality errors may occur.
- **Weighting:** All available target measurements, which comprise the set Σ_i , are used to update the persistent particle weights according to

$$w_{\text{pers}}^p(i) = \left[1 - p_D + \sum_{z_j \in \Sigma_i} w_{j,\text{update}}^p(i) \right] w_{\text{pers}}^p(i|i-1), \quad (32)$$

with

$$w_{j,\text{update}}^p(i) = \frac{p_D f_i(z_j | \mathbf{x}^p(i))}{(\lambda_{\text{FA}} C_{\text{FA}}(z_j) + \mathcal{L}(z_j)) C(z_j)}, \quad (33)$$

and

$$\mathcal{L}(z_j) = \sum_{q=1}^{N_{\text{pers}}(i)} p_D f_i(z_j | \mathbf{x}^q(i)) w_{\text{pers}}^q(i|i-1). \quad (34)$$

Note that – in contrast to the D-PPHDF – the weighting step is applied only once using the entire set of measurements. Therefore – and since there might be more than one measurement per target – we have to ensure that the weight update terms $w_{j,\text{update}}^p$ – and consequently the particle weights – still sum to the number of targets present. This is done by normalizing Equation (33) with $C(z_j)$, i.e., the cardinality of the cluster to which the current measurement z_j belongs.

Afterwards, we form the set $\Sigma_{i,\text{cand}}$ of candidate measurements for the ATB step according to

$$\Sigma_{i,\text{cand}} = \Sigma_i \setminus \left\{ z_{m_p} \mid m_p = \arg \max_j w_{j,\text{update}}^p(i), \right. \\ \left. p = 1, \dots, N_{\text{pers}}(i) \right\}. \quad (35)$$

- **Resampling:** The expected number of targets $\hat{N}_{\text{tgt}}(i)$ is calculated from the total persistent particle mass as

$$\hat{N}_{\text{tgt}}(i) = \left\lceil \sum_{p=1}^{N_{\text{tot}}(i)} w_{\text{pers}}^p(i) \right\rceil. \quad (36)$$

Consequently, the number of persistent particles is updated according to

$$N_{\text{pers}}(i) = \hat{N}_{\text{tgt}}(i) N_P. \quad (37)$$

Furthermore, the set of persistent particles is resampled by drawing $N_{\text{pers}}(i)$ particles with probability $\frac{w_{\text{tgt}}^p(i)}{\hat{N}_{\text{tgt}}(i)}$. Then, the weights are reset to equal values as

$$w_{\text{pers}}^p(i) = \frac{\hat{N}_{\text{tgt}}(i)}{N_{\text{pers}}(i)}, \quad p = 1, \dots, N_{\text{pers}}(i). \quad (38)$$

- **Clustering:** In contrast to the D-PPHDF, there is only one estimate of the expected number of targets. Hence, we can use *k-means clustering* [40] to find the estimated target states by grouping the resampled particles into $\hat{N}_{\text{tgt}}(i)$ clusters and calculating the centroid of each cluster.
- **Roughening:** Roughening is performed analogously to the D-PPHDF.
- **Adaptive Target Birth:** N_P new particles are placed randomly around each candidate measurement $z_j \in \Sigma_{i,\text{cand}}$ yielding a total number of $N_{\text{new}}(i) = N_P \cdot |\Sigma_{i,\text{cand}}|$ newborn particles. The corresponding weights are chosen according to

$$w_{\text{new}}^p(i) = \frac{p_B}{N_{\text{new}}(i)}, \quad p = 1, \dots, N_{\text{new}}(i), \quad (39)$$

where p_B is the probability of birth.

The pseudo-code of the MS-PPHDF is given in Table III.

B. Computational Complexity and Communication Load

In this section we analyze the computational complexity and the communication load of the MS-PPHDF. The following steps are performed at every time instant i but time dependency is omitted for simplicity:

- *Prediction*: The prediction step described by Equations (30) and (31) is performed for each of the N_{tot} particles and is linear in the dimensionality d .
 $\Rightarrow \mathcal{O}(N_{\text{tot}}d)$
- *Pre-Clustering*: The pre-clustering step relies on single-linkage clustering. The complexity is therefore cubic in the total number of measurements N_{meas} . [37]
 $\Rightarrow \mathcal{O}(N_{\text{meas}}^3)$
- *Weighting*: Each particle is updated in the weighting step given by Equations (32)–(34). The weight update as well as the designation of candidate measurements for ATB depends on the number of measurements $N_{\text{meas}} = |\Sigma|$.
 $\Rightarrow \mathcal{O}(N_{\text{tot}}N_{\text{meas}})$
- *Resampling*: The estimation of the number of targets and the resampling step in Equations (36)–(38) are linear in the number of particles used for the calculation [36].
 $\Rightarrow \mathcal{O}(N_{\text{tot}} + N_{\text{tgt}}N_P)$
- *Clustering*: In contrast to the D-PPHDF we can use k -means clustering. The complexity of Lloyd's implementation is given by [41]
 $\Rightarrow \mathcal{O}((N_{\text{tgt}}N_P)^{dN_{\text{tgt}}+1} \log(N_{\text{tgt}}N_P))$.
- *Roughening*: Roughening is linear in the dimensionality of the particles and their number.
 $\Rightarrow \mathcal{O}(N_{\text{tgt}}N_Pd)$
- *Adaptive Target Birth*: The birth process depends on the number of particles per target N_P as well as the number of candidate measurements N_{cand} .
 $\Rightarrow \mathcal{O}(N_PN_{\text{cand}})$

In summary, the computational complexity of the MS-PPHDF is largely comparable to that of the D-PPHDF. The only exception is the pre-clustering step, which scales cubically with the total number of measurements and adds additional complexity to the algorithm. As a tradeoff the communication load of the MS-PPHDF clearly is lower compared to the D-PPHDF because there is only the initial transmission of measurements from the nodes to the fusion center. However, considering a setup with relatively small communication radii, this initial communication step requires a lot of relaying and leads to high traffic density in the vicinity of the fusion center. Furthermore, this communication structure exhibits a single point of failure while a distributed sensor network is inherently redundant.

V. SIMULATIONS

In this section, we evaluate the performance of the D-PPHDF as well as the MS-PPHDF for tracking multiple targets in a sensor network with 1-coverage. In addition, we compare them to the alternative distributed PPHDF from [20], which will be referred to as Distributed Data Fusion Particle PHD Filter (DDF-PPHDF). Here, each node runs its own PPHDF using only its own measurements. In a subsequent step, the particles are distributed over the neighborhood and

reweighted by fusing their corresponding Exponential Mixture Densities.

Furthermore, we formulate the DPCRLB as a lower bound for evaluating the performance of the three algorithms in terms of the OSPA [15] metric. In our simulations, we compute the OSPA metric with respect to the joint set of target state estimates of the entire active network. The latter is found by clustering the target state estimates of all active nodes. Furthermore, we consider the squared OSPA metric scaled by the number of targets, i.e., $N_{\text{tgt}} \cdot \left(\bar{d}_p^{(c)}\right)^2$, as in [42]. That way, we can use the DPCRLB, which will be introduced in the following, as a benchmark.

A. The Distributed Posterior Cramér-Rao Lower Bound (DPCRLB)

Rather than evaluating the performance of the different MTT algorithms based on an error metric, it makes more sense to derive a minimum variance bound on the estimation error, which enables an absolute performance evaluation. For time-invariant statistical models, the most commonly used bound is the Cramér-Rao Lower Bound (CRLB), which is given by the inverse of Fisher's information matrix [43]. In [42] and [44], the CRLB is used in the context of multi-sensor MTT of an unknown number of unlabeled targets in order to evaluate the performance, as well as prove the asymptotic efficiency of the PHD as the number of nodes goes to infinity. Since we are more interested in the tracking behavior of a fixed network over time, we resort to the Posterior Cramér-Rao Lower Bound (PCRLB), which is an extension of the CRLB for the time-variant case [16]. This bound can be calculated sequentially with the help of a Riccati-like recursion derived in [45]. Furthermore, in [17] and [46], the PCRLB is adapted for an MTT scenario in which the tracker can obtain more than one measurement per target.

Let $\pi_i^m, m = 1, \dots, M$ denote the probability that any measurement is associated with target m at time instant i as defined in [18]. With the corresponding stochastic process Π_i^m , the new stochastic process of association probabilities and target states to be estimated becomes $\Phi_i = \left(\Pi_i^{1:M}, \Xi_i^{1:M}\right)$.

Fisher's information matrix $J_{\Phi_i} = \begin{bmatrix} J_{\Pi_i} & J_{\Pi_i \Xi_i} \\ J_{\Xi_i \Pi_i} & J_{\Xi_i} \end{bmatrix}$ can now be formed as described in [46] and [17]. However, as the number of targets varies over time, i.e., targets might enter or exit the ROI, J_{Φ_i} has to be expanded or shrunk in the inverse matrix domain as described in [47]. The PCRLB B_i at time instant i can be obtained as the trace of the inverted submatrix J_{Ξ_i} according to [46]

$$B_i = \text{trace} \left\{ \left[J_{\Xi_i} - J_{\Xi_i \Pi_i}^{-1} J_{\Pi_i} J_{\Pi_i \Xi_i} \right]^{-1} \right\}. \quad (40)$$

Note that, in a distributed MTT scenario, B_i corresponds to a lower bound on the estimation error of a central processing unit with access to all measurements. Since we are interested in completely distributed MTT with in-network processing, we extend the PCRLB to the Distributed Posterior Cramér-Rao Lower Bound (DPCRLB). To this end, each node k computes

its own PCRLB B_i^k considering only the measurements of its two-hop neighborhood, which is given by

$$\mathcal{N}_k^{(2)} = \bigcup_{l \in \mathcal{N}_k} \mathcal{N}_l, \quad (41)$$

i.e., the neighbors of node k and their neighbors. Furthermore, only the targets within the sensing range of \mathcal{N}_k are taken into account. Clearly, only nodes with a neighborhood in the vicinity of at least one target will be able to calculate a PCRLB. The DPCRLB $B_{i,\text{dist}}$ at time instant i is then obtained by averaging over these values according to

$$B_{i,\text{dist}} = \frac{1}{|\mathcal{M}|} \sum_{k \in \mathcal{M}} B_i^k, \quad (42)$$

where \mathcal{M} is the set of all nodes that are able to compute a PCRLB.

B. Simulation Setup

In the following simulations, a static sensor network as depicted in Figure 1a) is used to perform MTT. The network is centered around the point of origin $[0, 0]^\top$ and distributed such that 1-coverage of the ROI is guaranteed. It covers an area of approximately 2500 m². Clutter is assumed to be uniformly distributed over the sensing range of each node with an average rate of $\lambda_{\text{FA}} = 0.1$. For the sake of simplicity, collisions between targets and sensor nodes are neglected. An overview of all simulation parameters is given in Table I. Since the purpose of this work is to introduce the MS-PPHDF as well as the D-PPHDF, verify their functionality, and compare them to alternative approaches, we consider a rather simple scenario with a high probability of detection and a low clutter level. In our future work, we will study more sophisticated scenarios to define possible breakdown points of our algorithms.

We use the MS-PPHDF, the D-PPHDF, as well as the DDF-PPHDF to track three targets for $i = 0, \dots, 30$. The targets enter the ROI at time steps $i = 0, 9, 14$ from the north, south, and west, respectively. A Monte Carlo simulation with $N_{\text{MC}} = 1000$ runs is performed to evaluate the performance of the tracking algorithms in terms of the Optimal Subpattern Assignment (OSPA) metric and compute the DPCRLB as a benchmark. Note that the target trajectories as shown in Figure 1a) are deterministic, as is often the case in target tracking simulations [47] in order to guarantee the comparability of the different Monte Carlo runs regarding, for instance, the number of targets present.

C. Simulation Results

The simulation results for tracking three targets in the given setup are depicted in Figure 2. While the top part considers zero-mean Gaussian measurement noise with a per-component variance of $\sigma_r^2 = 0.1$, the bottom part shows the results for $\sigma_r^2 = 0.3$. In addition to evaluating the performance of MS-PPHDF, D-PPHDF, and DDF-PPHDF in terms of the squared and scaled OSPA metric over time and comparing it to the DPCRLB as can be seen in the left part of the figure, we also look at the estimated number of targets, which is depicted

TABLE I: Simulation parameters

VARIABLE	VALUE	DESCRIPTION
Δi	1 s	time step of the tracking algorithm
N	30	number of nodes
σ_r^2	0.1, 0.3 m ²	componentwise power of meas. noise
σ_q^2	0.01 m ²	componentwise power of state noise
R_{com}	$2R_{\text{sen}}$	communication radius
R_{sen}	6 m	sensing radius
E_c	6	empirical interval length for jitter
K	0.2	tuning constant for roughening
N_P	500	number of particles per target
p_B	0.8	probability of birth
p_D	0.95	probability of detection
p_S	0.98	probability of survival
λ_{FA}	0.1	average no. of false alarms / clutter
$c_{\text{FA}}(z)$	$\frac{1}{\pi R_{\text{sen}}^2}$	PDF of false alarms / clutter (uniform)
c	2	cut-off value (OSPA)
p	2	order of the OSPA metric

in the right column. Since the OSPA metric contains a penalty for an erroneous estimate of the number of targets, this side-by-side comparison facilitates the interpretation of the tracking results.

Let us start by considering Figures 2a) and 2b), i.e., the case of $\sigma_r^2 = 0.1$. First of all, we observe that neither tracking algorithm provides an OSPA value or an estimate of the number of targets for $i = 0$. This is expected and due to ATB, which initializes new particle clouds based on the measurements from the previous time step. Thus, target birth is delayed by one time step and tracking can only be performed for $i > 0$. The same effect can be witnessed at $i = 9$ and $i = 14$, respectively, which mark the time instants at which targets 2 and 3 enter the ROI. Here, the OSPA curves of all trackers exhibit a spike, which is due to the fact that the newborn particles are not yet considered in the tracker and, hence, the number of estimated targets is too low, as can be seen in Figure 2b).

Another sudden rise of all the OSPA curves can be observed in the time interval $20 \leq i \leq 24$ with a valley at $i = 22$. Looking at the estimated number of targets, we can attribute this phenomenon to the fact that only two of the three targets are recognized by the tracking algorithms. Since the target trajectories are deterministic, we know that in the given time interval targets 2 and 3 cross paths. Due to the inability of the clustering algorithm to separate strongly overlapping sets of measurements, the two targets merge into one as long as they are close to each other. When the two targets occupy almost exactly the same position, i.e., at $i = 22$, the OSPA metric decreases due to the decrease in measurement variance. As the targets drift apart, the variance and with it the OSPA metric increases up to the point where the two targets can be recognized as separate again and the corresponding penalty is switched off.

Looking at the overall picture in Figure 2a), which shows the case of $\sigma_r^2 = 0.1$, it is evident that the centralized MS-PPHDF and the distributed D-PPHDF achieve approximately the same performance with OSPA values closely

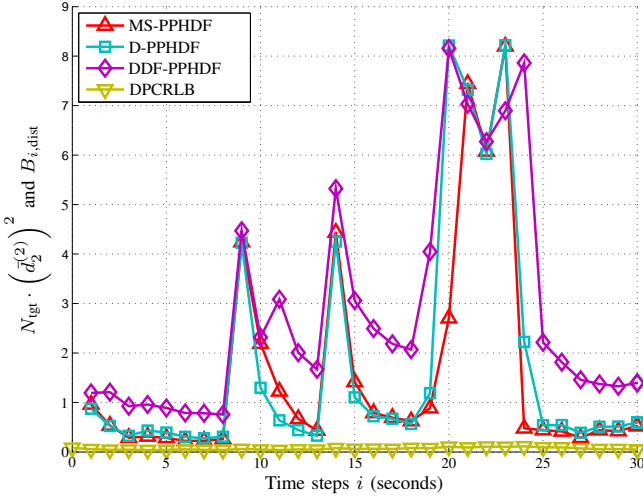
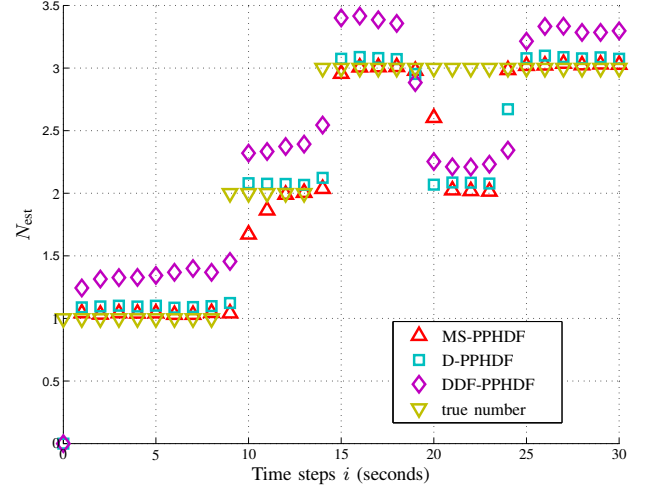
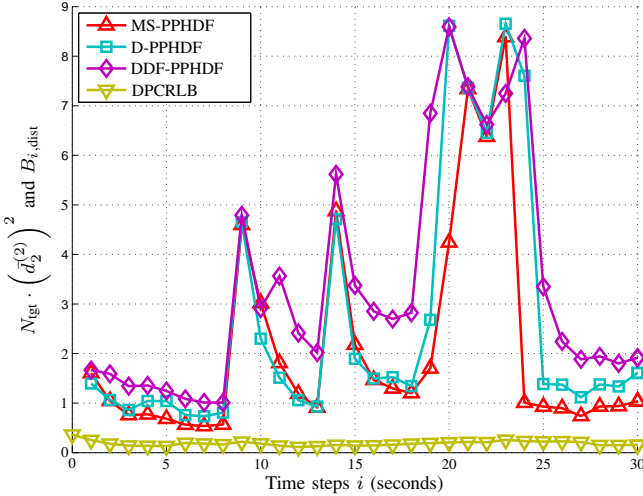
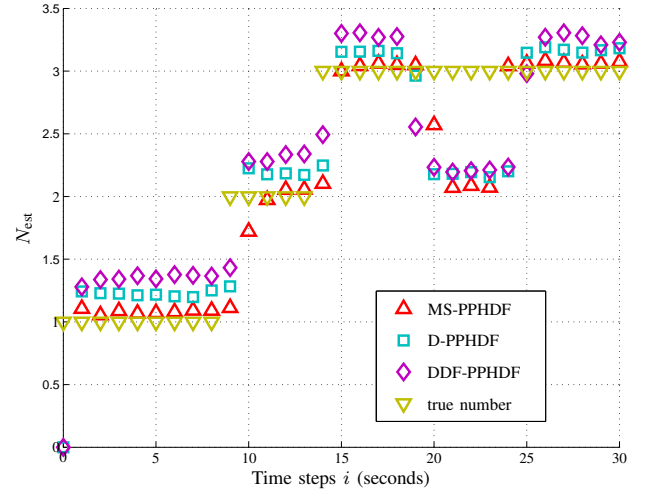
(a) Squared and scaled OSPA compared to DPCRLB, $\sigma_r^2 = 0.1$ (b) Estimated number of targets, $\sigma_r^2 = 0.1$ (c) Squared and scaled OSPA compared to DPCRLB, $\sigma_r^2 = 0.3$ (d) Estimated number of targets, $\sigma_r^2 = 0.3$

Fig. 2: Simulation results for tracking three targets in the ROI over time using the Multi-Sensor Particle PHD Filter (MS-PPHDF), the Diffusion Particle PHD Filter (D-PPHDF), and the Distributed Data Fusion Particle PHD Filter (DDF-PPHDF). The left part of the figure shows the squared and scaled Optimal Subpattern Assignment (OSPA) metric for each algorithm compared to the Distributed Posterior Cramér-Rao Lower Bound (DPCRLB), while the right part compares the estimated to the true number of targets.

approaching the DPCRLB when the number of targets stays constant. Furthermore, both algorithms deliver very accurate estimates of the number of targets, given they are separable by clustering, as can be seen in Figure 2b). The DDF-PPHDF, however, continuously exhibits a worse performance than the D-PPHDF, both in terms of the OSPA metric as well as the estimated number of targets. This is where the additional communication in the proposed D-PPHDF shows its strength in reducing uncertainty due to measurement noise and clutter. Apart from achieving worse tracking results, the DDF-PPHDF also has more difficulty in separating targets 1 and 2 when they cross paths, resulting in an earlier rise and a later fall of the OSPA metric, compared to our approach.

In the case of $\sigma_r^2 = 0.3$, the overall performance of the different tracking algorithms is very similar to the case of $\sigma_r^2 = 0.1$. In order to make a statement on how the different

tracking algorithms compare, let us neglect the penalty due to an erroneous estimate of the number of targets and take a look at Figures 3a) and 3b), which are zoomed-in versions of Figures 2a) and 2c), respectively.

In Figures 3a) and 3b) the DPCRLB is given as a benchmark for tracking performance. One can observe that its value is always smaller or equal to the respective measurement variance. As stated before, the centralized MS-PPHDF and the distributed D-PPHDF exhibit very similar performance and deliver better tracking results than the DDF-PPHDF. While the MS-PPHDF achieves lower OSPA values than the D-PPHDF when the number of targets stays constant, i.e., for $3 \leq i \leq 8$ and $24 \leq i \leq 30$, the D-PPHDF performs better directly after a new target appears, i.e., for $1 \leq i \leq 2$, $10 \leq i \leq 13$, and $15 \leq i \leq 18$. This is likely due to the fact that the two-step communication scheme employed in the D-PPHDF is able to

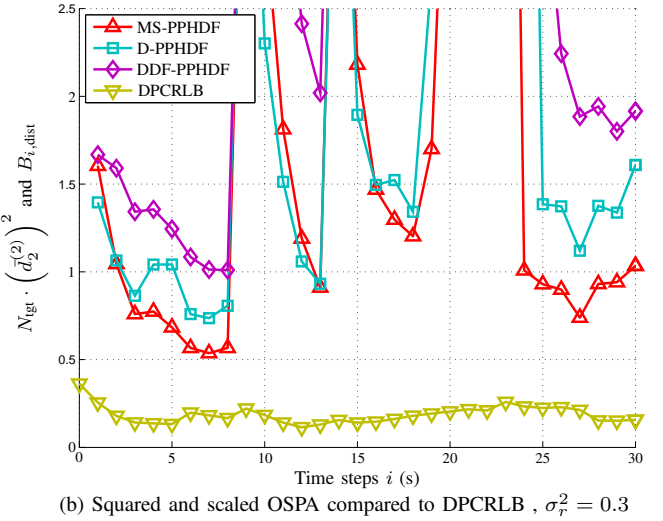
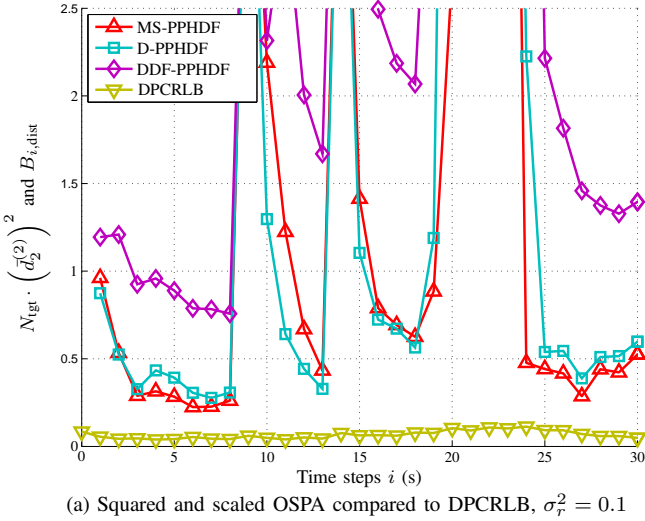


Fig. 3: Simulation results for tracking three targets in the ROI over time (zoomed in). The squared and scaled Optimal Subpattern Assignment (OSPA) metrics of using the Multi-Sensor Particle PHD Filter (MS-PPHDF), the Diffusion Particle PHD Filter (D-PPHDF), and the Distributed Data Fusion Particle PHD Filter (DDF-PPHDF), are compared to the Distributed Posterior Cramér-Rao Lower Bound (DPCRLB).

reduce the impact of measurement noise and clutter faster than the centralized MS-PPHDF can.

Looking at the case of $\sigma_r^2 = 0.3$ in Figure 3b), we observe that the higher measurement noise affects the performance of all algorithms, resulting in higher OSPA curves. While the OSPA curves of the MS-PPHDF and the DDF-PPHDF are proportionally shifted upward by approximately the same value, i.e., they are equally impacted by the higher noise level, the D-PPHDF seems to be slightly more affected by the change. However it still outperforms the DDF-PPHDF at all time instants.

All in all, the proposed D-PPHDF yields better performance than the existing DDF-PPHDF in estimating the number of targets and tracking them, irrespective of the amount of measurement noise. In addition, it is also a bit faster in delivering correct state estimates of new targets than the centralized MS-PPHDF and performs only slightly worse once the number of targets stays constant. In our future work, we will look at ways to further improve the performance of the MS-PPHDF and the D-PPHDF in order to approach the DPCRLB even more closely.

VI. CONCLUSION

In this work, we developed a distributed as well as a centralized PPHDF for MTT in sensor network. We, furthermore, came up with a distributed version of the PCRLB that served as a benchmark in the performance evaluation. Our simulation results showed that the distributed D-PPHDF is faster in correctly tracking new targets than the centralized MS-PPHDF and performs only slightly worse when the number of targets stays constant. In addition, it delivers accurate tracking results as long as the targets are far enough apart so that their corresponding measurement clouds are separable. Our approach outperforms the existing DDF-PPHDF at the cost of additional communication between sensor nodes.

ACKNOWLEDGMENT

The authors would like to thank Dr. Paolo Braca from the NATO Science & Technology Organization, Centre for Maritime Research and Experimentation in La Spezia, Italy, for his valuable comments.

REFERENCES

- [1] S. Challa, R. Evans, M. Morelande, and D. Musicki, *Fundamentals of object tracking*. Cambridge University Press, 2011.
- [2] S. Maresca, P. Braca, J. Horstmann, and R. Grasso, “Maritime surveillance using multiple high-frequency surface-wave radars,” *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 52, no. 8, pp. 5056–5071, Aug 2014.
- [3] J. Rambach, M. F. Huber, M. R. Balthasar, and A. M. Zoubir, “Collaborative multi-camera face recognition and tracking,” in *Proceedings of the 12th IEEE International Conference on Advanced Video- and Signal-based Surveillance (AVSS2015)*, August 2015.
- [4] R. Olfati-Saber, J. A. Fax, and R. M. Murray, “Consensus and cooperation in networked multi-agent systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, Jan 2007.
- [5] O. Hlinka, F. Hlawatsch, and P. M. Djuric, “Distributed particle filtering in agent networks: A survey, classification, and comparison,” *IEEE Signal Processing Magazine*, vol. 30, no. 1, pp. 61–81, 2013.
- [6] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, “Diffusion strategies for distributed Kalman filtering: Formulation and performance analysis,” *Proceedings of the 1st IAPR Workshop on Cognitive Information Processing (CIP)*, pp. 36–41, 2008.
- [7] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, “A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking,” *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.
- [8] Y. Bar-Shalom, P. K. Willett, and X. Tian, *Tracking and Data Fusion: A Handbook of Algorithms*. Storrs, CT: YBS Publishing, 2011.
- [9] D. Reid, “An algorithm for tracking multiple targets,” *IEEE Transactions on Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979.
- [10] S. Oh, L. Schenato, P. Chen, and S. Sastry, “Tracking and coordination of multiple agents using sensor networks: System design, algorithms and experiments,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 234–254, 2007.
- [11] D. E. Clark, “Multiple target tracking with the probability hypothesis density filter,” Ph.D. dissertation, Department of Electrical, Electronic and Computer Engineering, Heriot-Watt University, UK, October 2006.

1: **input:** $d, E_c, K, n, N, N_P, p_B, p_S, \lambda_{FA}, C_{FA}$

2: **initialize:** $\{s_{k,\text{coll}}^p(0), w_{k,\text{coll}}^p(0)\}_{p=1}^{N_{k,\text{coll}}(0)} = \{s_{k,\text{new}}^p(0), w_{k,\text{new}}^p(0)\}_{p=1}^{N_{k,\text{new}}(0)} = \emptyset$.

3: **while** $i \leq n$ **do**

4: **for** $k = 1, \dots, N$ **do**

5: Merge the sets of collective persistent and newborn particles with corresponding weights:

$$\{s_{k,\text{tot}}^p(i), w_{k,\text{tot}}^p(i)\}_{p=1}^{N_{k,\text{tot}}(i)} = \{s_{k,\text{coll}}^p(i-1), w_{k,\text{coll}}^p(i-1)\}_{p=1}^{N_{k,\text{coll}}(i-1)} \cup \{s_{k,\text{new}}^p(i-1), w_{k,\text{new}}^p(i-1)\}_{p=1}^{N_{k,\text{new}}(i-1)}.$$

6: **for** $p = 1, \dots, N_{k,\text{tot}}(i)$ **do**

7: Predict the new state of each particle and update the weight with the probability of survival p_S :

$$\begin{aligned} s_{k,\text{pers}}^p(i) &= \mathbf{F} s_{k,\text{tot}}^p(i) \\ w_{k,\text{pers}}^p(i|i-1) &= p_S w_{k,\text{tot}}^p(i). \end{aligned}$$

8: Update the weights using neighborhood measurements:

$$\begin{aligned} w_{k,\text{pers}}^p(i) &= \prod_{l \in \mathcal{N}_k} \left[1 - p_D + \sum_{\mathbf{z}_j \in \Sigma_i^l} w_{k,j,\text{update}}^p(i) \right] w_{k,\text{pers}}^p(i|i-1), \\ w_{k,j,\text{update}}^p(i) &= \frac{p_D f_i(\mathbf{z}_j | \mathbf{x}^p(i))}{\lambda_{FA} C_{FA}(\mathbf{z}_j) + \mathcal{L}(\mathbf{z}_j)}, \\ \mathcal{L}(\mathbf{z}_j) &= \sum_{q=1}^{N_{k,\text{pers}}(i)} p_D f_i(\mathbf{z}_j | \mathbf{x}^q(i)) w_{k,\text{pers}}^q(i|i-1). \end{aligned}$$

9: **end for**

10: Form the set of candidate measurements for ATB:

$$\Sigma_{i,\text{cand}}^k = \Sigma_i^k \setminus \left\{ \mathbf{z}_{m_p} \mid m_p = \arg \max_j w_{k,j,\text{update}}^p(i), \quad p = 1, \dots, N_{k,\text{pers}}(i), \quad j = 1, \dots, |\Sigma_i^l| \forall l \in \mathcal{N}_k \right\}.$$

11: Calculate the estimated number of targets:

$$\hat{N}_{k,\text{tgt}}(i) = \left\lceil \sum_{p=1}^{N_{k,\text{tot}}(i)} w_{k,\text{pers}}^p(i) \right\rceil.$$

12: Resample $\hat{N}_{k,\text{pers}}(i) = \hat{N}_{k,\text{tgt}}(i) N_P$ particles and reset the weights:

$$w_{k,\text{pers}}^p(i) = \frac{\hat{N}_{k,\text{tgt}}(i)}{N_{k,\text{pers}}(i)}, \quad p = 1, \dots, N_{k,\text{pers}}(i).$$

13: Merge the sets of persistent neighborhood particles and weights:

$$\{s_{k,\text{coll}}^p(i), w_{k,\text{coll}}^p(i)\}_{p=1}^{N_{k,\text{coll}}(i)} = \bigcup_{l \in \mathcal{N}_k} \{s_{l,\text{pers}}^p(i), w_{l,\text{pers}}^p(i)\}_{p=1}^{N_{l,\text{pers}}(i)}.$$

14: Use single-linkage clustering to identify $\hat{N}_{\text{tgt}}(i)$ clusters and find the set of estimated target states $\{\hat{\mathbf{s}}_k^l(i)\}_{l=1}^{\hat{N}_{\text{tgt}}(i)}$ by calculating the centroids.

15: Add an independent jitter to each particle using a component-wise standard deviation of:

$$\sigma_c^j(i) = K E_c N_{k,\text{coll}}(i)^{-1/d}.$$

16: Place N_P new particles randomly around each candidate measurement $\mathbf{z}_j \in \Sigma_{i,\text{cand}}^k$. Set the weights as:

$$w_{k,\text{new}}^p(i) = \frac{p_B}{N_{k,\text{new}}(i)}, \quad p = 1, \dots, N_{k,\text{new}}(i).$$

17: **end for**

18: $i \leftarrow i + 1$

19: **end while**

20: **return**

TABLE II: The Diffusion Particle PHD Filter.

1: **input:** $d, E_c, K, n, N, N_P, p_B, p_S, \lambda_{FA}, c_{FA}$

2: **initialize:** $\{s_{\text{coll}}^p(0), w_{\text{coll}}^p(0)\}_{p=1}^{N_{\text{coll}}(0)} = \{s_{\text{new}}^p(0), w_{\text{new}}^p(0)\}_{p=1}^{N_{\text{new}}(0)} = \emptyset$.

3: **while** $i \leq n$ **do**

4: Merge the sets of persistent and newborn particles with corresponding weights:

$$\{s_{\text{tot}}^p(i), w_{\text{tot}}^p(i)\}_{p=1}^{N_{\text{tot}}(i)} = \{s_{\text{pers}}^p(i-1), w_{\text{pers}}^p(i-1)\}_{p=1}^{N_{\text{pers}}(i-1)} \cup \{s_{\text{new}}^p(i-1), w_{\text{new}}^p(i-1)\}_{p=1}^{N_{\text{new}}(i-1)}.$$

5: **for** $p = 1, \dots, N_{\text{tot}}(i)$ **do**

6: Predict the new state of each particle and update the weight with the probability of survival p_S :

$$\begin{aligned} s_{\text{pers}}^p(i) &= \mathbf{F} s_{\text{tot}}^p(i) \\ w_{\text{pers}}^p(i|i-1) &= p_S w_{\text{tot}}^p(i). \end{aligned}$$

7: Cluster the measurements using single-linkage clustering and assign each measurement \mathbf{z} a label $C(\mathbf{z})$ reflecting the cardinality of its cluster.

8: Update the weights using the measurements of the entire network:

$$\begin{aligned} w_{\text{pers}}^p(i) &= \left[1 - p_D + \sum_{\mathbf{z}_j \in \Sigma_i} w_{j,\text{update}}^p(i) \right] w_{\text{pers}}^p(i|i-1) \\ w_{j,\text{update}}^p(i) &= \frac{p_D f_i(\mathbf{z}_j | \mathbf{x}^p(i))}{(\lambda_{FA} c_{FA}(\mathbf{z}_j) + \mathcal{L}(\mathbf{z}_j)) C(\mathbf{z}_j)}, \\ \mathcal{L}(\mathbf{z}_j) &= \sum_{q=1}^{N_{\text{pers}}(i)} p_D f_i(\mathbf{z}_j | \mathbf{x}^q(i)) w_{\text{pers}}^q(i|i-1). \end{aligned}$$

9: **end for**

10: Form the set of candidate measurements for ATB:

$$\Sigma_{i,\text{cand}} = \Sigma_i \setminus \left\{ \mathbf{z}_{m_p} \mid m_p = \arg \max_j w_{j,\text{update}}^p(i), \quad p = 1, \dots, N_{\text{pers}}(i), \quad j = 1, \dots, |\Sigma_i| \right\}.$$

11: Calculate the estimated number of targets:

$$\hat{N}_{\text{tgt}}(i) = \left\lceil \sum_{p=1}^{N_{\text{tot}}(i)} w_{\text{pers}}^p(i) \right\rceil.$$

12: Resample $\hat{N}_{\text{pers}}(i) = \hat{N}_{\text{tgt}}(i) N_P$ particles and reset the weights:

$$w_{\text{pers}}^p(i) = \frac{\hat{N}_{\text{tgt}}(i)}{N_{\text{pers}}(i)}, \quad p = 1, \dots, N_{\text{pers}}(i).$$

13: Find the set of estimated target states $\{\hat{s}^l(i)\}_{l=1}^{\hat{N}_{\text{tgt}}(i)}$ by using k -means clustering and calculating the centroid of each cluster.

14: Add an independent jitter to each particle using a component-wise standard deviation of:

$$\sigma_c^j(i) = K E_c N_{\text{pers}}(i)^{-1/d}.$$

15: Place N_P new particles randomly around each candidate measurement $\mathbf{z}_j \in \Sigma_{i,\text{cand}}$. Set the weights as:

$$w_{\text{new}}^p(i) = \frac{p_B}{N_{\text{new}}(i)}, \quad p = 1, \dots, N_{\text{new}}(i).$$

16: $i \leftarrow i + 1$

17: **end while**

18: **return**

TABLE III: The Multi-Sensor Particle PHD Filter.

- [12] R. Mahler, "Multitarget Bayes filtering via first-order multitarget moments," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1152–1178, 2003.
- [13] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill, "Integrated coverage and connectivity configuration in wireless sensor networks," in *Proceedings of the 1st international conference on Embedded networked sensor systems*. ACM, 2003, pp. 28–39.
- [14] M. R. Balthasar, S. Al-Sayed, S. Leier, and A. M. Zoubir, "Optimal area coverage in autonomous sensor networks," in *Proceedings of the 2nd International Conference and Exhibition on Underwater Acoustics (UA2014)*, June 2014.
- [15] D. Schuhmacher, B.-T. Vo, and B.-N. Vo, "A consistent metric for performance evaluation of multi-object filters," *Signal Processing, IEEE Transactions on*, vol. 56, no. 8, pp. 3447–3457, Aug 2008.
- [16] H. L. Van Trees, *Detection, estimation, and modulation theory*. John Wiley & Sons, 2004.
- [17] C. Hue, J.-P. Le Cadre, and P. Perez, "Performance analysis of two sequential Monte Carlo methods and posterior Cramer-Rao bounds for multi-target tracking," in *Proceedings of the Fifth International Conference on Information Fusion*, vol. 1, 2002, pp. 464–473.
- [18] —, "Sequential Monte Carlo methods for multiple target tracking and data fusion," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 309–325, 2002.
- [19] M. Uney, D. Clark, and S. Julier, "Distributed fusion of PHD filters via exponential mixture densities," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 3, pp. 521–531, June 2013.
- [20] M. Uney, S. Julier, D. Clark, and B. Ristic, "Monte Carlo realisation of a distributed multi-object fusion algorithm," in *Sensor Signal Processing for Defence (SSPD 2010)*, Sept 2010, pp. 1–5.
- [21] G. Battistelli, L. Chisci, C. Fantacci, A. Farina, and A. Graziano, "Consensus CPHD filter for distributed multitarget tracking," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 7, no. 3, pp. 508–520, 2013.
- [22] R. Mahler, "The multisensor PHD filter: I. General solution via multi-target calculus," in *Proc. SPIE Signal Processing, Sensor Fusion, and Target Recognition XVIII*, vol. 7336, May 2009.
- [23] —, "Approximate multisensor CPHD and PHD filters," in *Proceedings of the 13th Conference on Information Fusion (FUSION)*, July 2010, pp. 1–8.
- [24] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund, "Particle filters for positioning, navigation, and tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 425–437, 2002.
- [25] B. T. Vo, "Random finite sets in multi-object filtering," Ph.D. dissertation, University of Western Australia, 2008.
- [26] B.-N. Vo, S. Singh, and A. Doucet, "Sequential Monte Carlo implementation of the PHD filter for multi-target tracking," in *Proceedings of the 6th International Conference on Information Fusion*, 2003, pp. 792–799.
- [27] —, "Sequential Monte Carlo methods for multitarget filtering with random finite sets," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 4, pp. 1224–1245, 2005.
- [28] R. Mahler, "Multitarget moments and their application to multitarget tracking," *Technical Report, DTIC Document*, 2001.
- [29] —, "'Statistics 102' for multisource-multitarget detection and tracking," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 3, pp. 376–389, 2013.
- [30] B. Ristic, D. Clark, B.-N. Vo, and B.-T. Vo, "Adaptive target birth intensity for PHD and CPHD filters," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 2, pp. 1656–1668, April 2012.
- [31] B. Ristic, D. Clark, and B.-N. Vo, "Improved SMC implementation of the PHD filter," in *Proceedings of the 13th Conference on Information Fusion (FUSION)*, July 2010, pp. 1–8.
- [32] S. Hong, L. Wang, Z.-G. Shi, and K. S. Chen, "Simplified particle PHD filter for multiple-target tracking: Algorithm and architecture," *Progress In Electromagnetics Research*, vol. 120, pp. 481–498, 2011.
- [33] A. H. Sayed, "Diffusion adaptation over networks," in *E-Reference Signal Processing*, R. Chellappa and S. Theodoridis, Eds. New York: Elsevier, 2013.
- [34] B. Everitt, S. Landau, M. Leese, and D. Stahl, *Cluster Analysis*. John Wiley & Sons, 2011.
- [35] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," in *IEE Proceedings F – Radar and Signal Processing*, vol. 140, no. 2, 1993, pp. 107–113.
- [36] F. Gustafsson, "Particle filter theory and practice with positioning applications," *IEEE Aerospace and Electronic Systems Magazine*, vol. 25, no. 7, pp. 53–82, 2010.
- [37] F. Murtagh, "A survey of recent advances in hierarchical clustering algorithms," *The Computer Journal*, vol. 26, no. 4, pp. 354–359, 1983.
- [38] K. Granstrom and U. Orguner, "A PHD filter for tracking multiple extended targets using random matrices," *Signal Processing, IEEE Transactions on*, vol. 60, no. 11, pp. 5657–5671, Nov 2012.
- [39] K. Granstrom, A. Natale, P. Braca, G. Ludeno, and F. Serafino, "PHD extended target tracking using an incoherent X-band radar: Preliminary real-world experimental results," in *Information Fusion (FUSION), 2014 17th International Conference on*, July 2014, pp. 1–8.
- [40] J. MacQueen et al., "Some methods for classification and analysis of multivariate observations," in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, no. 281–297. California, USA, 1967, p. 14.
- [41] M. Inaba, N. Katoh, and H. Imai, "Applications of weighted voronoi diagrams and randomization to variance-based k-clustering: (extended abstract)," in *Proceedings of the 10th Annual Symposium on Computational Geometry*, ser. SCG '94. New York, NY, USA: ACM, 1994, pp. 332–339. [Online]. Available: <http://doi.acm.org/10.1145/177424.178042>
- [42] P. Braca, S. Marano, V. Matta, and P. Willett, "Asymptotic efficiency of the PHD in multitarget/multisensor estimation," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 3, pp. 553–564, June 2013.
- [43] S. M. Kay, *Fundamentals of statistical signal processing: estimation theory*. Prentice-Hall, Inc., 1993.
- [44] P. Braca, S. Marano, V. Matta, and P. Willett, "Multitarget-multisensor ML and PHD: Some asymptotics," in *Proceedings of the 15th International Conference on Information Fusion (FUSION)*, 2012, pp. 2347–2353.
- [45] P. Tichavsky, C. Muravchik, and A. Nehorai, "Posterior Cramer-Rao bounds for discrete-time nonlinear filtering," *IEEE Transactions on Signal Processing*, vol. 46, no. 5, pp. 1386–1396, 1998.
- [46] C. Hue, J.-P. Le Cadre, and P. Pérez, "Performance analysis of two sequential Monte Carlo methods and posterior Cramér-Rao bounds for multi-target tracking," IRISA, Tech. Rep., 2002.
- [47] A. Bessell, B. Ristic, A. Farina, X. Wang, and M. Arulampalam, "Error performance bounds for tracking a manoeuvring target," in *Proceedings of the 6th International Conference on Information Fusion (FUSION)*, vol. 2, 2003, pp. 903–910.



Mark R. Balthasar (S'14) received his B.Sc. and M.Sc. degrees in electrical engineering and information technology with summa cum laude from Technische Universität Darmstadt, in 2011 and 2014, respectively. He is currently working towards his Ph.D. degree in the Signal Processing Group, Institute of Telecommunications, Technische Universität Darmstadt. His research interest lies in collaborative estimation as well as sequential detection in autonomous sensor networks.

Mark R. Balthasar received the *Young Scientist Award for the Best Paper Presented by a Graduate Student* (second place) at the 2nd International Conference and Exhibition on Underwater Acoustics (UA2014) in 2014.



Abdelhak M. Zoubir (F'08) is a Fellow of the IEEE and IEEE Distinguished Lecturer (Class 2010-2011). He received his Dr.-Ing. from Ruhr-Universität Bochum, Germany in 1992. He was with Queensland University of Technology, Australia from 1992-1998 where he was Associate Professor. In 1999, he joined Curtin University of Technology, Australia as a Professor of Telecommunications. In 2003, he moved to Technische Universität Darmstadt, Germany as Professor of Signal Processing and Head of the Signal Processing Group. His research interest

lies in statistical methods for signal processing with emphasis on bootstrap techniques, robust detection and estimation and array processing applied to telecommunications, radar, sonar, automotive monitoring and safety, and biomedicine. He published over 400 journal and conference papers on these areas.

Professor Zoubir served as General Chair and Technical Chair of numerous international IEEE conferences, most recently he was the Technical Co-Chair of ICASSP-14 held in Florence, Italy. Dr. Zoubir also served on publication boards of various journals, most notably, he served a three-year term as Editor-In-Chief of the IEEE Signal Processing Magazine (2012-2014). Dr. Zoubir was Chair (2010-2011), Vice-Chair (2008-2009) and Member (2002-2007) of the IEEE SPS Technical Committee Signal Processing Theory and Methods (SPTM). He was a Member of the IEEE SPS Technical Committee Sensor Array and Multi-channel Signal Processing (SAM) from 2007 until 2012. He also serves on the Board of Directors of the European Association of Signal Processing (EURASIP) (2009-2016) and on the Board of Governors of the IEEE SPS (2015-2017). He is the President-Elect of EURASIP, starting his term in January 2017.